Binary Decision Diagrams: A Mathematical Model for the Path-Related Objective Functions

P.W.C. PRASAD
College of Information Technology,
United Arab Emirates University, UAE

ALI ASSI
Department of Electrical Engineering,
United Arab Emirates University, UAE

BRUCE MILLS
College of Information Technology,
United Arab Emirates University, UAE

Abstract -This paper describes a mathematical model for all path length parameters (APL: Average Path Length, LPL: Longest Path Length, and SPL: Shortest Path Length) of Binary Decision Diagrams (BDDs). The proposed model is based on an empirical analysis of randomly generated Boolean functions. The formal core of the developed model is a unique equation for the path-related objective functions over the set of BDDs derived from Boolean functions with given number of variables and Sum of Products (SOP) terms. Simulation results show good correlation between the theoretical results and those predicted by the mathematical model. This model provides an estimation of the performance of a circuit prior to its final implementation, and can be applied to Boolean functions with any number of variables, any number of product terms, and any variable ordering method.

Key-Words: Evaluation time estimation, Binary Decision Diagram (BDD), Path Length of BDDs, Boolean functions

1 Introduction

Advanced Computer-Aided Design (CAD) systems are essential tools in the design of modern Very Large Scale Integration (VLSI) circuits. One of the most important functions of CAD tools is to provide robust and efficient data structures to represent Boolean functions as well as fast algorithms to manipulate these data structures. During the last two decades, BDDs have achieved great popularity as data structures in the synthesis and verification of digital systems.

BDD in general is a direct acyclic graph representation of Boolean functions initially proposed by Bryant and Akers [1]. Analysis of the complexity of Boolean functions can be performed through the BDDs that represent these functions [2].

Evaluation of time complexity of Boolean functions using BDDs is one of the important factors in the evaluation of logic circuits, which is an important step in the design of digital systems and the verification of their efficiency [3]. The time complexity of Boolean functions is proportional to the path length of their BDDs. Therefore, minimization of the path length can improve the overall performance of the circuit implementing Boolean functions [5], [6]. Many research works have analyzed the behavior of path-related objective functions [4], [5], [6]. Among these works, the minimization of the number of paths [9] which is an important task in the test of BDD circuits, and the minimization of Disjoint SOPs [7] used in computing the spectra of Boolean functions. The methods proposed for the APL minimization are based on either Static variable ordering [8] or dynamic variable ordering techniques [9]. The minimization of APL leads to circuits with smaller depth of paths from the root to the terminal node of the BDD. The resulting circuit will be optimized for speed on one hand and on the other hand the number of very long paths in the BDD will be reduced [10]. The minimization of APL is of great importance in embedded systems, real time operating system applications [11].

Minimization of the LPL [4] and the SPL in BDDs is motivated by the synthesis of digital circuits in order to optimize their delays, which is an important task in Pass Transistor Logic (PTL) [12], [13]. The minimization of Minimum Path Length (MPL) has been discussed in [9]. One of the main problems with PTL is the presence of long paths: the delay of a chain of n pass transistors is proportional to n^2. Inserting buffers can reduce the path length, but this increases the silicon area. Consequently the minimization of the longest path will definitely improve the performance of the resulting circuit in term of speed [12].

In all path length minimizations we need to create the whole BDD representing the Boolean function with the best possible variable ordering. Building the whole BDD may lead to an increase in the time complexity of the design process. It will be useful to have an estimation of the BDD complexity prior to build it [14], [15].

Therefore, it is important to develop a mathematical model that predicts the path length of a BDD. Over the past two decades most of the problems in the synthesis, design and testing of combinational circuits have been solved using various mathematical methods [16]. For any combinational circuit the only available initial information is the Boolean function that represents this circuit and the number of its variables. This information is usually considered to design and verify circuits using well known mathematical methods. A mathematical model to predict the complexity of Boolean functions, XOR/XNOR min-terms and the path length of BDDs using empirical fit were introduced in [17], [18], [19], [20], and [21].

The main objective of this paper is to introduce an improved mathematical model of path-related objective functions compared to the methods proposed in [22]. The proposed model will provide exact parameters for the path length complexity for any number of variables and for any variable ordering method which allows the analysis of circuit performance without building its BDD. The model is also capable of predicting the number of product terms in the Boolean function that leads to the maximum time complexity. The remaining of this paper is divided as follows: Section two provides a brief explanation of the mathematical model based on an empirical analysis of the path length. The proposed method with simulation results followed by the mathematical model are given in section three and four respectively. Finally, in section five we conclude this research work with a summary of our future developments.

2 Mathematical model based on Empirical analysis of Path length behavior

The work in [21] improves the methodologies proposed in [5] and [19] for the estimation of path length complexities. Analysis of the time complexity in BDDs based on SPL and APL is presented in [20]. For each variable count n between 1 and 14 inclusive and for each term count between 1 and 2^n-1, 100 SOP terms were randomly generated and the Colorado University Decision Diagram (CUDD) package [31] was used to determine the SPL and APL. This process was repeated until the average size of the SPL and APL complexities (i.e. number of nodes) became 1. Then the simulated graphs for APL and SPL complexities (Figure 1 and figure 2) were plotted against the product term count for each number of variables.

Analysis of the APL and SPL graphs shows that the path length complexity may be modeled by an expression of the form of equation (1):

$$complex\,(t) = \sum_{i=1}^{2}\left(\frac{\log(t+1)}{(\log(t+1))^{\beta_i}}\right)^{\alpha_i} \qquad (1)$$

where, t is the number of product terms in the Boolean function, α and β are constants that shape the peak of the graph. We determined the α and β as follows:

For the SPL, $\alpha_1 = 7, \beta_1 = 1$ and $\alpha_2 = 10$ lead to a close fit, $\beta_2 = 3$ for a number of variables v less than 11 $(v \le 11)$ and $\beta_2 = 5$ for $v \ge 12$. The following equation (7) was used in order to compute β_2 for SPL,

$$\beta_2 = 3 + \left(\frac{1.8}{[e^{(v-11.5)}]^2 + 1}\right) \qquad (2)$$

For the APL, $\alpha_1 = 7, \beta_1 = 0.7$ and $\alpha_2 = 10$ lead to a close fit, $\beta_2 = 2.1$ for $v \le 11$ and $\beta_2 = 3.5$ for $v \ge 12$. The following equation (3) was used in order to compute constant β_2 for APL,

$$\beta_2 = 0.7\left(3 + \left(\frac{1.8}{[e^{(v-11.5)}]^2 + 1}\right)\right) \qquad (3)$$

Finally, the APL and SPL curves can be obtained using the following single equation (4):

$$Complex\,(t,v) = \sum_{i=1}^{2} \cdot y_i \left(\frac{\log\left(t/x_i + 1 \right)}{\left(\log\left(t/x_i + 1 \right) \right)^{\beta_i\,(v)}} \right)^{\alpha_i\,(v)} \qquad (4)$$

In this mathematical model, the peaks (x_i, y_i) for both the APL and SPL curves were found by performing an empirical fit.

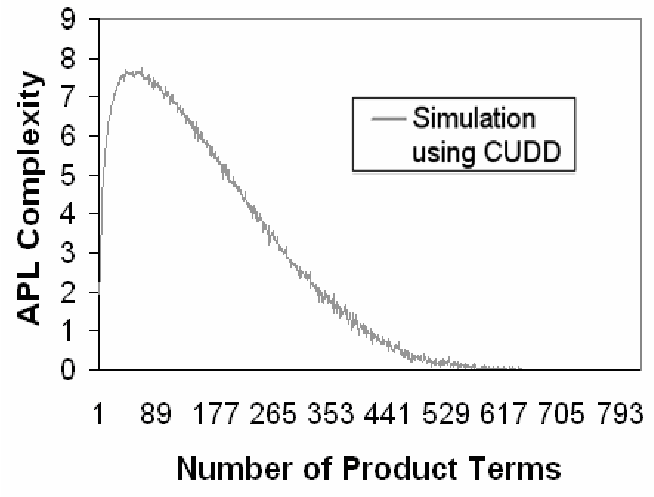

Fig. 1: APL complexity variation for 10 Variables

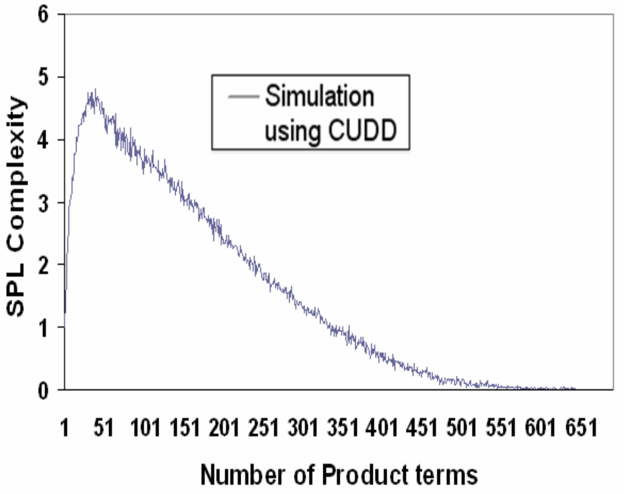

Fig. 2: SPL complexity variations for 10 Variables

3 LPL, SPL and APL Complexities

The mathematical model explained in the previous section is repeated here in order to find the graphs for the path-related objective functions (i.e. LPL, SPL and APL) complexities. The graphs of LPL, SPL and APL will be plotted against the product term count for each number of variables.

Figures 1, 2 and 3 illustrate the APL, SPL and LPL complexities for Boolean functions with product terms having n=10 variables and using the Symmetric Sift reordering technique of the CUDD tool. Due to its efficiency compared to other reordering techniques of the CUDD, the Symmetric Sift reordering technique is used here as the base variable ordering method. The graphs indicate that the complexity of the path length in general (LPL, SPL and APL) increases as the number of product terms increases. This is clear from the rising edge of the curve shown in Figures 1, 2 and 3. At the end of the rising edge in the graphs, the size of the APL, LPL and SPL reaches a maximum ($APL \cong 7.735$, $LPL \cong 10$, and $SPL \cong 5.4$ in this case). This peak indicates the maximum APL, LPL and SPL complexities that any Boolean function with 10 variables can have independently of the number of product terms. Apart of that the peak also specifies the number of product terms (critical limit) of a Boolean function that leads to the maximum complexities of APL, LPL and SPL for any Boolean function with 10 variables.

The number of product terms that leads to the maximum complexities of APL, LPL and SPL is 66, 17 and 50 respectively. If the number of product terms increases above the critical limit, as expected, the product terms starts to simplify and the BDD will reduce, which will reduce the path lengths size. The APL, SPL and LPL complexity graphs shown in Figures 1, 2 and 3 indicate that as the number of product terms increases the complexity of the APL, SPL, and LPL decreases in a slower rate and ultimately reaches 0.

Figure 3 shows that the LPL graph behaves a bit different than the other complexity graphs shown in Figures 1 and 2 where the overt features of the curve, an initial sharp rise, a peak, a plateau, and a roll-off were observed to be independent of the variable count. The location and height of the peak, the width and height of the plateau and the slope of the logarithm of the roll-off varied.

Even though the rising edge of the SPL complexity variation graph is similar to the APL graph, slope of the logarithm of the roll-off varied and might have more than one peak in the roll-off. The location and height of the peak and the slope of the logarithm of the roll-off varied. Reduction of the APL, SPL and LPL complexities implies that all the product terms simplify to logic 1. A simple algebraic expression for these curves is developed, unifying all the cases.

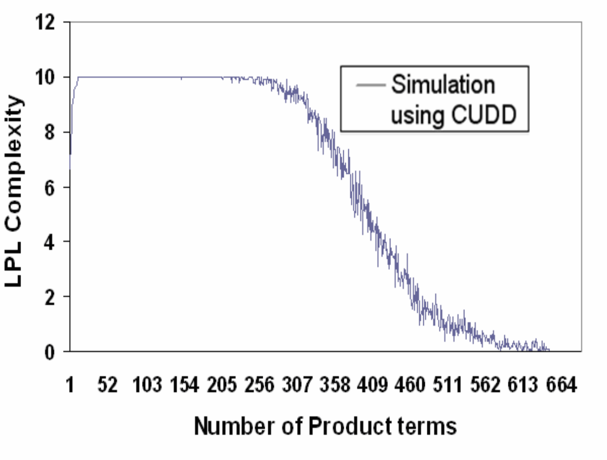

Fig. 3: LPL complexity variation for 10 Variables

4 Mathematical model for the Path length Complexities

A large variety of curves with closed forms (for example: a power series with enough terms) can be fitted to the empirical APL, SPL, and LPL curves as shown in figures 4-6. But, if the curve does not reflect the real behavior, it is likely that the number of parameters will increase at least linearly with the number of curves being fitted. A model [28] developed by the same authors had exactly this property, for each variable count, a curve was fitted requiring around 6 parameters per variable. The model presented here has two c-parameters, and two s-parameters for each class of curve (LPL, APL, and SPL) rather than each variable count. The fitting is global and requires at most 6 parameters per class of curves. The model presented here is definitely empirical, but based on a theoretical justification of possibility being developed. The current work of the authors includes the firming and extending of this argument as far as possible. Part of this argument suitable for publication is included here for completeness.

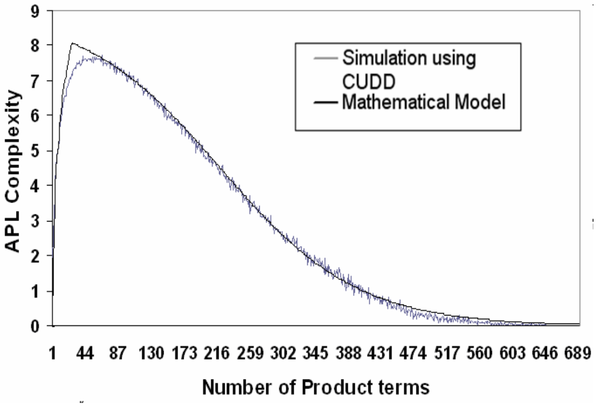

Fig. 4: Simulation/Mathematical APL complexity variation for 10 Variables

Three styles of curve fitting are illustrated. For LPL, the fitting of a sigmoidal curve to the roll-off, showing how the choice of parameters varies linearly with the variable count.

For the APL, the logarithmic rise is also fitted, showing that its essential shape does not change with the variable count, but the peak shape is not attempted. For the SPL, the entire curve, including the peak, is fitted. This will result in some less intuitive algebraic behavior, but a closer fit to the real behavior.

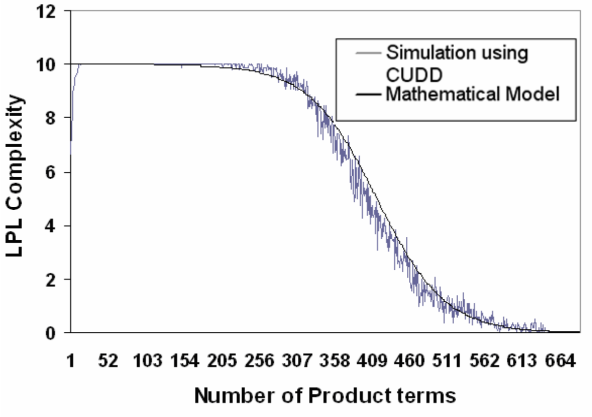

Fig. 5: Simulation/Mathematical LPL complexity variation for 10 Variables

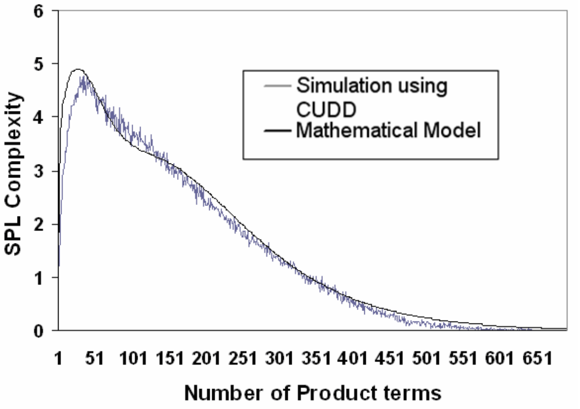

Fig. 6: Simulation/Mathematical SPL complexity variation for 10 Variables

For each class of curves, either the minimum or the product of three functions with parameters affine in v is indicated. The functions are $\log(x)$ for the rise (Figure 7), $e^{(-x)} \cdot e^{-(e^{-x})^2}$ for the peak and $1/(1+e^{x})$ for the roll-off, where the second function is the composition of $e^{(-x)}$ and $e^{(-\cdot x^2)}$. Each target function is modeled by a function similar to (5), for lower values of *t* and function (6) for higher values of *t*, where *t* is the number of product terms, *v* the number of variables and the vertical scaling *a*(*v*), horizontal shift $c(v)$ and the horizontal scaling $s(v)$ are affine functions of v. The parameter $(2/3)^v$ in function (6) defines the global scaling that normalizes all the curves and $g(t,c,s) = 1/(1+e^{(t-c)/s})$.

$$\log_{1.5}(t+1) \quad (5)$$

$$a(v)\cdot g(t\cdot(2/3)^v, c(v), s(v)) \quad (6)$$

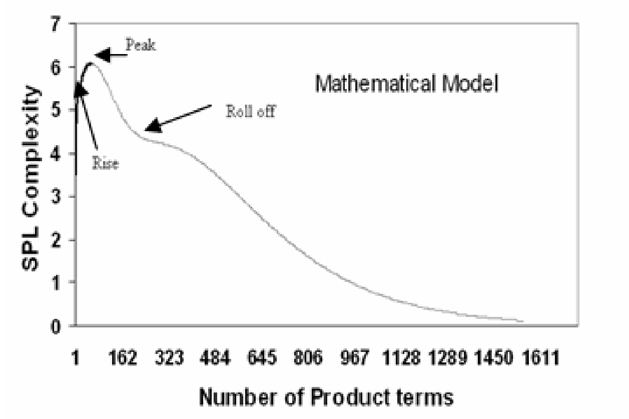

Fig. 7: Mathematical fit for the SPL complexity behavior

It can be seen from figures 4-6 that the mathematical models match well the curves obtained by simulation, with relatively simple functions, and relatively few parameters. All path lengths (i.e. APL, SPL, and LPL) are fitted globally with at most 6 parameters for each type, rather than the fitting of the empirical analysis [20] that had a similar number of parameters for each value of v. This is a major improvement in this mathematical model.

Further verification of the mathematical model is done for Boolean functions with 2 to 14 variables. It can be inferred that the simulated and mathematical curves are following similar patterns for any number of variables. Figures 8-13 illustrates the simulated and mathematical models for APL, SPL and LPL for variables 13 and 8 respectively.

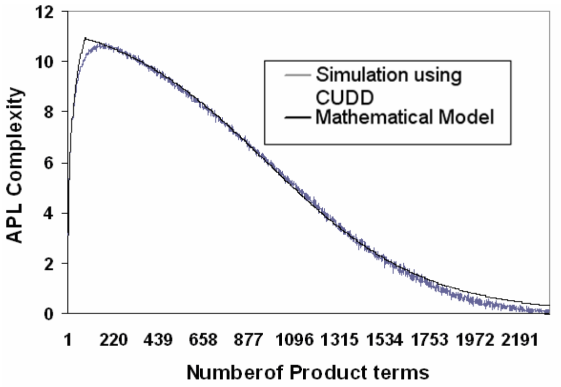

Fig. 8: Simulation/Mathematical APL complexity variation for 13 Variables

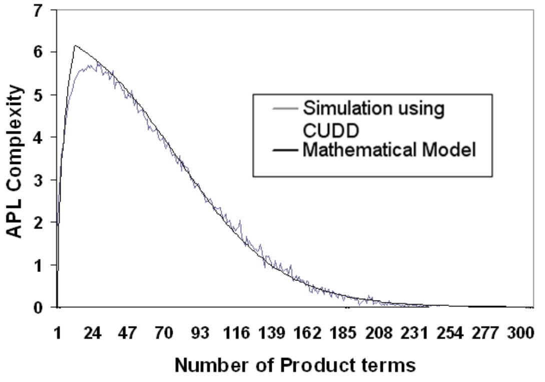

Fig. 9: Simulation/Mathematical APL complexity variation for 8 Variables

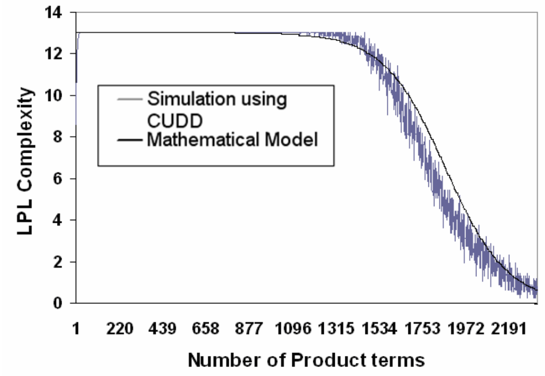

Fig. 10: Simulation/Mathematical SPL complexity variation for 13 Variables

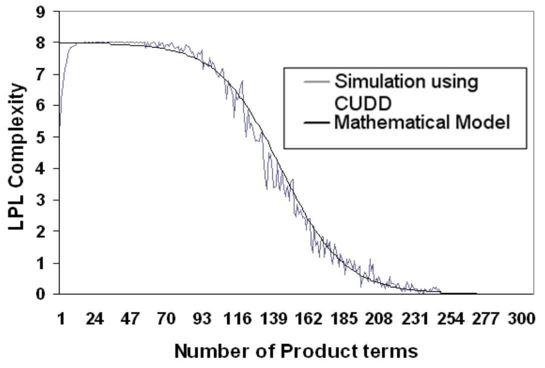

Fig. 11: Simulation/Mathematical LPL complexity variation for 8 Variables

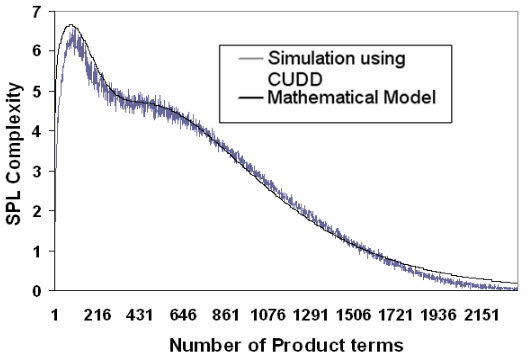

Fig. 12: Simulation/Mathematical LPL complexity variation for 13 Variables

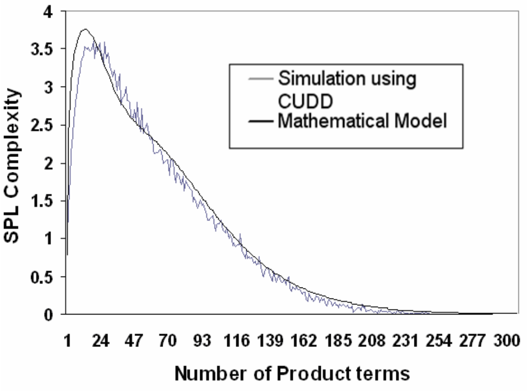

Fig. 13: Simulation/Mathematical LPL complexity variation for 8 Variables

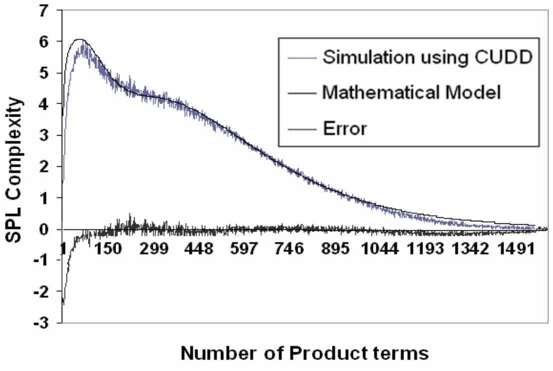

Fig. 14: SPL complexity error variation for 12 Variables

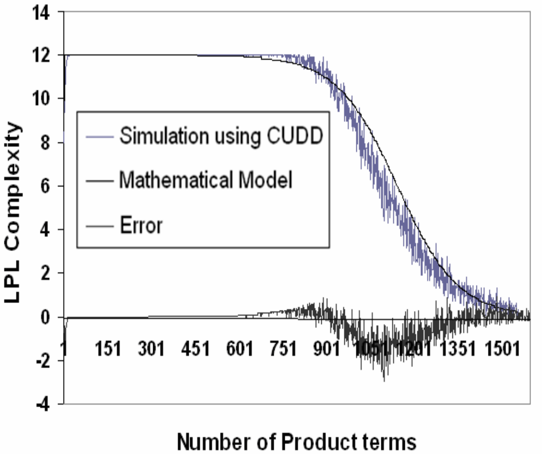

Fig. 15: LPL complexity error variation for 12 Variables

Figure 14-16 shows the efficiency of the proposed mathematical model, which produces complexity estimation error for SPL, LPL and APL**.** It can be inferred that the mathematical expression was able to match the simulated curve with minimum error, which is less then ± 0.1 for most of the product terms.

However, after cleaning up the curves a bit algebraically, our next step is to try to get some numerical measures, such as max absolute error, to be significantly smaller.

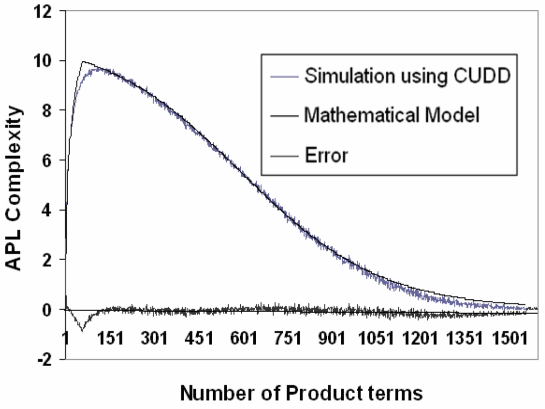

Fig. 16: APL complexity error variation for 12 Variables

5. Conclusion

We have discussed in this paper the idea of using BDDs to model a relationship between the path-related objective functions; namely the APL, SPL, and LPL, and the number of product terms in Boolean functions. Analyzing the simulation results, we have introduced a simple and unique mathematical model with the theoretical justification for the whole process. The three curves for APL, SPL and LPL were fitted with slightly different methods, but with only a few global parameters. Future work will be mainly concentrated on having wider range of *variables* to check that the affine fitting is correct and to verify the proposed method with real benchmark circuits.

References

[1] S. B. Akers, Binary Decision Diagram, *IEEE Trans. Computers*, Vol. 27, 1978, pp. 509-516.

[2] R. E. Bryant, Graph–Based Algorithm for Boolean Function Manipulation, *IEEE Trans. Computers*, Vol. 35, 1986, pp. 677-691.

[3] C. Scholl, R. Drechsler, and B. Becker, Functional simulation using binary decision diagrams, *Proceedings International Conference of CAD*, 1997, pp. 8-12.

[4] S. Nagayama, and T. Sasao, On the minimization of longest path length for decision diagrams, *Proc International Workshop on Logic and Synthesis (IWLS-2004),* 2004, pp. 28-35.

[5] P.W. C. Prasad, M. Raseen, A. Assi, and S. M. N. A. Senanayake, BDD Path Length Minimization based on Initial Variable Ordering *Journal of Computer Science,*

Science Publications, Vol. 1(4), 2005, pp. 521-529.

[6] R. Ebendt, and R. Drechsler, On the Exact Minimization of Path-Related Objective Functions for BDDs, *Proc. of Inter. Conf. on Very Large Scale integration (IFIP VLSI-SOC)*, 2005, pp. 525-530.

[7] N. Drechsler, M. Hilgemeier, G. Fey, and R. Drechsler, Disjoint Sum of Product Minimization by Evolutionary Algorithms, *Applications of Evolutionary Computing, Evo Workshops*, 2004, pp. 198-207.

[8] S. Nagayama, A. Mishchenko, T. Sasao, and J.T. Butler, Minimization of average path length in BDDs by variable reordering, *Intl. Workshop on Logic and Synthesis*, 2003, pp. 207-213.

[9] R. Rudell, Dynamic Variable Ordering for Ordered Binary Decision Diagrams *Proc. of the Inter. Conf. on Computer Aided Design (ICCAD)*, 1993, pp. 42-47.

[10] G. Fey, J. Shi and R. Drechsler, BDD Circuit Optimization for Path Delay Fault-Testability *Proc. of EUROMICRO Symposium on Digital System Design*, 2004, pp. 168-172.

[11] M. Lindgren, H. Hansson, and H. Thane, Using Measurements to Derive the Worst-case Execution Time *Proc. of 7th Inter. Conf. on Real-Time Systems and Applications (RTCSA'00)*, 2000, pp. 15-22.

[12] V. Bertacco, S. Minato, P. Verplaetse, L. Benini, and G. De Micheli, *Decision Diagrams and Pass Transistor Logic Synthesis*, Stanford University CSL Technical Report, No. CSL-TR-97-748, Dec. 1997.

[13] R. S. Shelar and S. S. Sapatnekar, Recursive Bi-partitioning of BDD's for Performance Driven Synthesis of Pass Transistor Logic, *Proc. of IEEE/ACM ICCAD*, 2001, pp. 449-452.

[14] N. Ramalingam, and S. Bhanja, Causal Probabilistic Input Dependency Learning for Switching model in VLSI Circuits, *Proc. of ACM Great Lakes Symposium on VLSI*, 2005, pp. 112-115.

[15] P. E. Dunne, and W. van der Hoeke, Representation and Complexity in Boolean Games *Proc. 9th European Conf. on Logics in Artificial Intelligence*, LNCS 3229, Springer-Verlag, 2004, pp. 347-355.

[16]. C. A. J. van Eijk, *Formal Methods for the Verification of Digital Circuits, PhD thesis*, Eindhoven University of Technology, Netherlands, 1997.

[17] M. Raseen, P.W.C. Prasad and A. Assi, Mathematical Model to Predict the Number of Nodes in an ROBDD, *The 47th IEEE Inter. Midwest Symposium on Circuit and Systems (MWSCAS)*, Vol. III, 2004, pp. 431-434.

[18] M. Raseen, P. W. C. Prasad, and A. Assi, An Efficient Estimation of the ROBDD's Complexity, *accepted for Publication in Integration - the VLSI journal*, Elsevier Publication, May 2005.

[19] P.W. C. Prasad , M. Raseen and S. M. N. A. Senanayake, XOR/XNOR Functional Behavior on ROBDD Representation, *Proc. of The 14th IASTED Inter. Conf. on Applied Simulation and Modeling*, 2005, pp. 115-119.

[20] P.W. C. Prasad , M. Raseen, A. Assi, S. M. N. A. Senanayake, and B.I. Mills, Evaluation time estimation for Pass Transistor Logic Circuits, accepted for publication in the proceedings in *3rd IEEE Inter. Workshop on Electronic Design, Test and Applications (DELTA'06)*, January 2006, pp. 422-428.

[21] A. Assi, P.W. C. Prasad , B. Mills, and A. El-Chouemi, Empirical Analysis and Mathematical Representation of the Path Length Complexity in Binary Decision Diagrams, *Journal of computer Science*, Science Publications, Vol. 2(3), 2005, pp. 236-244.

[22] P.W. C. Prasad, B.I. Mills, M. Raseen, and A. Assi, Evaluation time of Boolean Functions: An Estimation methodology based on the path length of their BDD Representations, *The 2004 Inter. Research Conf. on Innovations in Information Technology (IIT 2004)*, Dubai, 2005, pp. 1-10.

[23] F. Somenzi, "CUDD: CU Decision Diagram Package," *ftp://vlsi.colorado.edu/ pub/*, 2003.